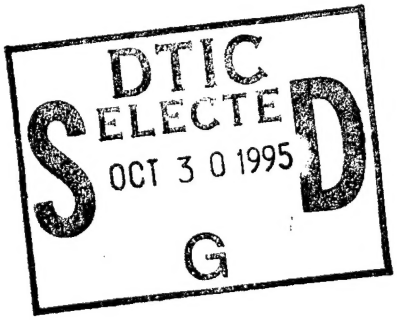
# Real-Time Architecture

SFRC. N00014-93-1-0611

FINAL REPORT

December 1994




Prepared for:

Department of the Navy
Office of Naval Research
800 North Quincy Street
Arlington, Virginia 22217-5000

Prepared By:

Jay K. Strosnider, Principal Investigator
Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-6927]

Principal Investigator Name:
Jay K. Strosnider (412) 268-6927 strosnider@ece.cmu.edu
PI Institution: Carnegie Mellon University
Contract title: Real-Time Architecture
Contract Number: N00014-93-1-0611

## 1 Productivity Measures

- Refereed papers submitted but not yet accepted: 4
- Refereed papers accepted and in press: 4
- Refereed papers published: 4
- Books submitted or published: none
- Other reports: 2
- Ph.D. dissertations: 3
- Masters dissertations: 1
- Patents filed or granted: none
- Invited presentations: 4
- Contributed presentations: 4
- Honors, Prizes and Awards received:
- Graduate students supported: 2
- Post-docs supported: 0
- Minorities supported: 0

Principal Investigator Names:
Jay K. Strosnider (412) 268-6927 strosnider@ece.cmu.edu
PI Institution: Carnegie Mellon University
Contract title: Real-Time Architecture
Contract Number: N00014-93-1-0611

# 2 Summary of Technical Progress

## 2.1 Overview

The research effort this year focused in the area of developing a unified framework for reasoning about timing correctness of standards-based resources. The intent is to develop systems engineering methodologies that support the composition of standards-based distributed, real-time systems with fully predictable timing properties. Further, we worked on developing theoretic foundations to integrate real-time scheduling and fault-tolerance domains.

The project members are also concerned with the transition of the theory being developed to the user community, especially the U.S. Navy. For this reason, project members have a close relationship with the RTSIA (Real-Time Scheduling in Ada) project in the CMU Software Engineering Institute.

Signifcant progress has been made. The development of real-time scheduling algorithms which support time redundancy is a difficult problem. Meeting the timing requirements of a real-time application often competes with the objective of allocating redundant time to meet the deadlines of recovery operations. All too often, the problem is skirted by designing schedulers which guarantee that the timing requirements of the real-time application workload are met, {\sl as long as no faults occur.} Unfortunately, the occurrence of faults and the subsequent need for recovery is a reality which must not be overlooked.

To address this problem this research was carried out with the following goals: \begin{tabbing} \hspace*{0.50in}\={\em 1.} \= to define a set of principles to guide the design of scheduling\\ \> \> methodologies for time redundancy management, \\ \> {\em 2.} \> to develop an analytical foundation to characterize and validate\\ \> \> the design alternatives, and \\ \> {\em 3.} \> to develop scheduling solutions which not only exhibit good performance \\ \> \> but are also implementable. \\ \end{tabbing} \vspace*{-0.20in} The following paragraphs describe our progress at achieving these goals.

The fundamental attributes of real-time systems which distinguish them from general-purpose computing systems were clearly articulated. These are the closely related properties of {\em predictability} and {\em schedulability}. A real-time workload is schedulable if its timing requirements are guaranteed to be met when executed on a predictable system. Every schedulable real-time workload has an associated amount of processing time the workload can be increased and still be schedulable. This amount of time is directly related to the quantity of redundant time available in a workload and is referred to as {\em slack}. The measurement of slack can take on various forms, differing in the volume and accuracy of the data required for its characterization. An effective use of slack for time redundancy requires the identification of a reasonable tradeoff between the overhead of managing large data structures and the accuracy needed to support predictable allocation.

Analytical methods to assess the schedulability of real-time workloads were developed for the case in which the tasks are scheduled according to a fixed-priority scheduling algorithm. These analytical

Carnegie Mellon

**Office of Sponsored Research**
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, Pennsylvania 15213-3890
412-268-2091

March 10, 1995

Defense Technical Information Center
Building 5, Cameron Station
Alexandria, Virginia 22304-6145

Subject: N00014-93-1-0611 - Final Technical Report

Dear DTIC Representative:

Enclosed please find a copy of the final technical report for the subject contract.

If you have any questions or require additional information, please contact me.

Sincerely,

Maryann Brendel

Maryann Brendel
Research Administrator

Enclosure

methods were extended to support the computation of slack. Although most of the ideas explored are applicable to a wide range of real-time system specifications, the primary focus of this research was restricted to the class of real-time workloads which are mission-critical and primarily consist of a set of independent periodic application tasks.

Two different strategies were introduced for manipulating the slack in a real-time workload to allocate redundant time for servicing fault recovery operations. These strategies were categorized as {\em static} and {\em dynamic} allocation strategies. Static allocation or preallocation strategies reserve redundant processing time for fault recovery off-line, while dynamic allocation strategies allocate time for recovery on-line.

Two static allocation algorithms for fixed-priority systems were proposed. The {\em Private Reservation Algorithm} (PRA) reserves time which is bound to the recovery of individual tasks. The {\em Communal Reservation Algorithm} (CRA) reserves a pool of time available to recovery operations on a first-come first-serve basis.

It was assumed that the allocation of recovery time should be fair in that there should be no explicit biases towards the recovery of some tasks over others, where possible. It was shown that at high workload utilizations some discriminatory treatment may be needed to ensure that the schedulability of the task set is not violated during recovery. Discrimination policies were discussed for each algorithm.

Closed-form expressions were developed for both algorithms to measure the reduction in processing capacity available to the fault-free workload after preallocation. The expressions allow us to evaluate the amount of processing time required to support various preallocation requirements.

Results showed that the cost of dedicating time for the recovery of individual tasks is usually much higher than the cost of sharing preallocated recovery time. For the particular case of guaranteed recovery from any single task failure, the PRA requires that 50\% of the processing capacity be preallocated for recovery, while an estimated 14\% is needed by the CRA for real-time workloads of 10 tasks or greater.

The dynamic allocation of time for recovery was described as a special case of a general problem in the scheduling theory literature known as the joint scheduling problem. The joint scheduling problem refers to the scheduling of periodic and aperiodic tasks on a common resource. Most research on joint scheduling has focused on the scheduling of hard periodic tasks and soft aperiodic tasks. However, hard aperiodic tasks better represent the timing characteristics of fault recovery operations. Although some researchers have recently addressed the joint scheduling problem when the aperiodic tasks are hard, they have focused only on dynamic-priority real-time systems. As a result, solutions for fixed-priority systems had not yet been proposed.

The {\em Slack Stealing Algorithm} was introduced for the joint scheduling of hard periodic and soft aperiodic tasks on fixed-priority systems. It is based on the approach of {\em slack stealing}, in which time for servicing aperiodic tasks is allocated by "stealing" all the processing time possible from the periodic tasks without causing their deadlines to be missed. This is equivalent to "stealing slack" from the periodic tasks.

The algorithm was proved to be optimal in providing the largest amount of high priority execution time for aperiodic processing while meeting all deadlines for the periodic tasks. The slack stealing algorithm was evaluated and compared against current methods for soft aperiodic service such as the sporadic server. Results showed that the slack stealing algorithm not only outperforms current aperiodic service

methods, but yields, in some circumstances, aperiodic response times which are essentially equal to those of an M/M/1 queue with the periodic workload ignored, even when the periodic load is very high. This is important because it makes it possible to use standard queueing formulas for computing mean aperiodic response times in some circumstances.

With these encouraging results, the slack stealing approach was extended to handle the scheduling of hard aperiodic tasks. It was demonstrated that there are important differences between the hard and soft aperiodic scheduling problems, namely, the unattainability of strong optimality and issues concerning the selection of the priority level used for aperiodic service.

Unfortunately, in some cases the slack stealing method introduces a large memory and scheduling overhead. A direct extension of slack stealing for the hard aperiodic scheduling case yields a worst-case scheduling overhead of O($n^{3}$), where $n$ is the number of periodic tasks.

To reduce the implementation overhead of the slack stealing method, an algorithm called the {\em Myopic Slack Management} (MSM) algorithm was introduced. Although the MSM algorithm is also based on the concept of slack stealing, the memory overhead is reduced by using conservative estimates of the slack available for each periodic task at run-time. To make slack estimation computationally feasible, the accumulation of available slack is restricted to relatively short intervals of time. As a result, it is said that the MSM algorithm is nearsighted, or myopic, in its ability to accumulate slack. The scheduling overhead is reduced by restricting the service of hard aperiodic tasks to a maximum of two priority levels; that of the failed periodic task issuing the recovery request, and the deadline monotonic priority level for the aperiodic. These techniques reduced the memory and scheduling overheads to a worst-case complexity of O($n$).

The performance of the MSM algorithm may be lower than that expected from a direct implementation of the slack stealing approach because of its tendency to underestimate the slack available and the limitations imposed on the priority levels considered for service. However, the MSM trades off some of the performance of the slack stealing method for a scheduling solution which has significantly less overhead.

A quantitative comparison of the performance of the static and dynamic allocation strategies was performed. Specifically, the comparison studies included the static Private and Communal Reservation Algorithms and the dynamic Myopic Slack Management algorithm.

To measure the effectiveness of an allocation algorithm, a metric referred to as {\em recovery coverage} was introduced. Recovery coverage parallels the well-known concept of error detection coverage. It was empirically computed as the percentage of recovery requests accepted for service relative to the total number of recovery requests issued during a simulation. Under no conditions was the service of a recovery request allowed to jeopardize the timing correctness of any fault-free application task.

Analytical models for predicting the coverage provided by the PRA and the CRA were derived. The predicted coverage for the PRA was shown to match the empirical results very closely. The prediction model for the CRA was shown to be optimistic but it offered insights in explaining the performance behavior of this algorithm.

All simulation results obtained for the application workloads considered were consistent. The MSM algorithm proved to be very robust to changes in periodic loading conditions and to increases in the size

of the transient recovery load. The preallocation algorithms rarely came close to providing the high coverage observed for the MSM algorithm.

Although the coverage estimates for the PRA remained stable as the size of the transient recovery load was increased, its coverage was highly sensitive to the periodic load. The CRA, on the other hand, was less sensitive to changes in the periodic load but its performance degraded significantly as the size of the recovery load was increased. In general, the performance of these preallocation algorithms was competitive with that of the MSM algorithm only in cases in which the joint processing load was small.

Most of the reported coverage values represent steady-state performance estimates, that is, estimates of the coverage provided by an algorithm when the transient recovery load is observed to persist for an infinite period of time. Since transient recovery loads only exist for short periods of time, the sensitivity of coverage to finite transient durations was investigated. Results showed that although coverage tends to increase as the duration of the transient decreases, the rate of change is very small. Hence, steady-state coverage values can be considered adequate approximations to the coverage observed for finite-duration transients, albeit slightly conservative.

The analytical framework was then extended to include run-time overheads introduced by the implementation of scheduling mechanisms in operating systems. The sources of run-time overheads in real-time operating systems were first identified. The times of their occurrence were referred to as {\em scheduling events}, which consist of the arrivals and completions of periodic or aperiodic tasks. By characterizing the work that the OS is required to service at each scheduling event, all run-time overheads affecting the timing behavior of real-time tasks were characterized. The work required at each scheduling event was expressed in terms of a proposed set of primitive OS functions. The combination of these primitive terms was used to estimate higher level implementation overheads such as task preemption and completion costs.

General scheduling models including overhead parameters were proposed. These models are useful for exploring the sensitivity of various timing properties of the workload to a variety of OS implementations. The worst-case execution times of the CRA, PRA, and MSM algorithms were determined to obtain overhead estimates with practical meaning. The overhead of supporting slack management operations in the MSM algorithm was also measured. To obtain an accurate representation of a real-time operating system, OS overhead data collected in related work for the RT Mach OS running on the same execution platform was used.

By using this set of representative OS overheads, a set of simulations of one real-time application was conducted. Estimates of the coverage provided by the Private and Communal Reservation Algorithms as well as the Myopic Slack Management Algorithm show a significant reduction in the coverage provided by these algorithms. However, the relative performance merits of each algorithm remained equal to those identified under ideal execution conditions.

Principal Investigator Names:
Jay K. Strosnider (412) 268-6927 strosnider@ece.cmu.edu
PI Institution: Carnegie Mellon University
Contract title: Real-Time Architecture
Contract Number: N00014-93-1-0611

## 3 Publications and Presentations

The following are project publications which were written, appeared or are in press for the above period. Project members made many research presentations during the contract period, and these are not enumerated. Each of the articles listed below that was published in a conference proceedings volume was presented at that conference.

### 3.1 Published or In Press

- J.K. Strosnider, J.P. Lehoczky and L. Sha, Deferrable Server Algorithm for Improved Asynchronous Response Times, to appear in IEEE Transactions on Computers.
- J.E. Sasinowski and J.K. Strosnider, A Dynamic Programming Algorithm for Cache/Memory Partitioning for Real-Time Systems, to appear in IEEE Transactions on Computers.
- D. Katcher, H. Arakawa and J.K. Strosnider, Engineering and Analysis of Fixed Priority Schedulers, IEEE Transactions on Software Engineering, September, 1993.
- S.S. Sathaye, L. Sha and J.K. Strosnider, Analysis of Reservation Based Dual Link Networks for Real-Time Applications, to appear in IEEE Transactions on Computers.
- S.S. Sathaye, A Yee, R.P. Bianchini, and J.K. Strosnider, A Distributed Routing Algorithm for Real-Time Traffic in Multi-Hop Neworks, submitted to IEEE Transactions on Parallel and Distributed Systems.
- S.S. Sathaye and J.K. Strosnider, Conventional & Early Token Release Schedluling Models for the IEEE 802.5 Token Ring, submitted to the International Journal on Real-Time Computing.
- S.S. Sathaye, D. Katcher and J.K. Strosnider, Fixed Priority Scheduling with Limited Priority Levels, submitted to IEEE Transactions on Computers.
- J.K. Strosnider and C.J. Paul, A Structured View of Real-Time Problem Solving, submitted to AI Magazine.
- D. Katcher and J.K. Strosnider, Dynamic versus Fixed Priority Scheduling: A Case Study, submitted to IEEE Transactions on Software Engineering.
- S.S. Sathaye, W.S. Kish and J.K. Strosnider, Responsive Aperiodic Services in High Speed Networks, Proceedings of the 13th International Conference on Distributed Computing, May 1993.
- H. Arakawa, D.I. Katcher, J.K. Strosnider and H.Tokuda, Modeling and Validation of the Real-Time Mach Scheduler, Proceedings of the 1993 Sigmetrics Conference on Measurement and Modeling of Computer Systems.
- S.S. Sathaye, L. Sha and J.K. Strosnider, Scheduling Real-Time Communications on Dual Link Networks, Proceedings of the 13th IEEE Real-Time Systems Symposium, December 1992.

### 3.2 Submitted for Publication

- D. Katcher, H. Arakawa and J.K. Strosnider, Engineering and Analysis of Fixed Priority Schedulers, IEEE Transactions on Software Engineering, minor revisions, resubmitted June, 1992.
- S.S. Sathaye, A Yee, R.P. Bianchini, and J.K. Strosnider, A Distributed Routing Algorithm for Real-Time Traffic in Multi-Hop Networks, IEEE Transactions on Parallel and Distributed Systems, Submitted April, 1992.
- S.S. Sathaye, L. Sha and J.K. Strosnider, Analysis of Reservation Based Dual Link Networks for Real-Time Applications, IEEE Transactions on Computers, Submitted April, 1992.
- S.S. Sathaye and J.K. Strosnider, A Scheduling Analysis of the IEEE 802.5 Token Ring With and Without Early Token Release, Real-Time Systems Journal, Submitted September, 1992.

### 3.3 Ph.D. Dissertations

- Ronald Mraz, "A RISC-Based Architecture for Real-Time Computation" Department of Electrical and Computer Engineering, Carnegie Mellon University, May, 1992.

### 3.4 Masters Dissertations

- Dwo-Wen Yan, "Dynamic Scheduling/Pacing System Prototype", Carnegie Mellon University, December, 1991.
- Trung Diep, "Performance Analysis of Superscalar Processors", Carnegie Mellon University, December 1991.
- Dan Katcher, "Engineering and Analysis of Fixed Priority Schedulers", Carnegie Mellon University, May, 1992.
- John Sasinowski, "A Dynamic Programming Algorithm for Cache/Memory Partitioning for Real-Time Systems", Carnegie Mellon University, May, 1992.

Principal Investigator Names:
Jay K. Strosnider (412) 268-6927 strosnider@ece.cmu.edu
PI Institution: Carnegie Mellon University
Contract title: Real-Time Architecture
Contract Number: N00014-93-1-0611

## 4 Transitions and DoD Interactions

Jay Strosnider interacts frequently with NOSC San Diego Distributed Combat Control project transitioning technology into Navy lab testbeds in San Diego. He also interacts with IBM, Bellcore and Intel on commercial applications of the developed technologies.

OFFICE OF THE UNDER SECRETARY OF DEFENSE (ACQUISITION)
DEFENSE TECHNICAL INFORMATION CENTER
CAMERON STATION
ALEXANDRIA, VIRGINIA 22304-6145

IN REPLY REFER TO DTIC-OCC

SUBJECT: Distribution Statements on Technical Documents

TO: OFFICE OF NAVAL RESEARCH
CORPORATE PROGRAMS DIVISION
ONR 353
800 NORTH QUINCY STREET
ARLINGTON, VA 22217-5660

1995 1027 053

1. Reference: DoD Directive 5230.24, Distribution Statements on Technical Documents, 18 Mar 87.

2. The Defense Technical Information Center received the enclosed report (referenced below) which is not marked in accordance with the above reference.

**FINAL REPORT**
**N00014-93-1-0611**
**TITLE: REAL-TIME ARCHITECTURE**

3. We request the appropriate distribution statement be assigned and the report returned to DTIC within 5 working days.

4. Approved distribution statements are listed on the reverse of this letter. If you have any questions regarding these statements, call DTIC's Cataloging Branch, (703) 274-6837.

FOR THE ADMINISTRATOR:

1 Encl

GOPALAKRISHNAN NAIR
Chief, Cataloging Branch

FL-171
Jul 93

DISTRIBUTION STATEMENT A:

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED

DISTRIBUTION STATEMENT B:

DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES ONLY; (Indicate Reason and Date Below). OTHER REQUESTS FOR THIS DOCUMENT SHALL BE REFERRED TO (Indicate Controlling DoD Office Below).

DISTRIBUTION STATEMENT C:

DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES AND THEIR CONTRACTORS; (Indicate Reason and Date Below). OTHER REQUESTS FOR THIS DOCUMENT SHALL BE REFERRED TO (Indicate Controlling DoD Office Below).

DISTRIBUTION STATEMENT D:

DISTRIBUTION AUTHORIZED TO DOD AND U.S. DOD CONTRACTORS ONLY; (Indicate Reason and Date Below). OTHER REQUESTS SHALL BE REFERRED TO (Indicate Controlling DoD Office Below).

DISTRIBUTION STATEMENT E:

DISTRIBUTION AUTHORIZED TO DOD COMPONENTS ONLY; (Indicate Reason and Date Below). OTHER REQUESTS SHALL BE REFERRED TO (Indicate Controlling DoD Office Below).

DISTRIBUTION STATEMENT F:

FURTHER DISSEMINATION ONLY AS DIRECTED BY (Indicate Controlling DoD Office and Date Below) or HIGHER DOD AUTHORITY.

DISTRIBUTION STATEMENT X:

DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES AND PRIVATE INDIVIDUALS OR ENTERPRISES ELIGIBLE TO OBTAIN EXPORT-CONTROLLED TECHNICAL DATA IN ACCORDANCE WITH DOD DIRECTIVE 5230.25, WITHHOLDING OF UNCLASSIFIED TECHNICAL DATA FROM PUBLIC DISCLOSURE, 6 Nov 1984 (Indicate date of determination). CONTROLLING DOD OFFICE IS (Indicate Controlling DoD Office).

The cited documents has been reviewed by competent authority and the following distribution statement is hereby authorized.

A

(Statement)

OFFICE OF NAVAL RESEARCH
CORPORATE PROGRAMS DIVISION
ONR 353
800 NORTH QUINCY STREET
ARLINGTON, VA 22217-5660

(Controlling DoD Office Name)

(Reason)

(Controlling DoD Office Address, City, State, Zip)

DEBRA T. HUGHES
DEPUTY DIRECTOR
CORPORATE PROGRAMS OFFICE

(Signature & Typed Name)

(Assigning Office)

19 SEP 1995

(Date Statement Assigned)